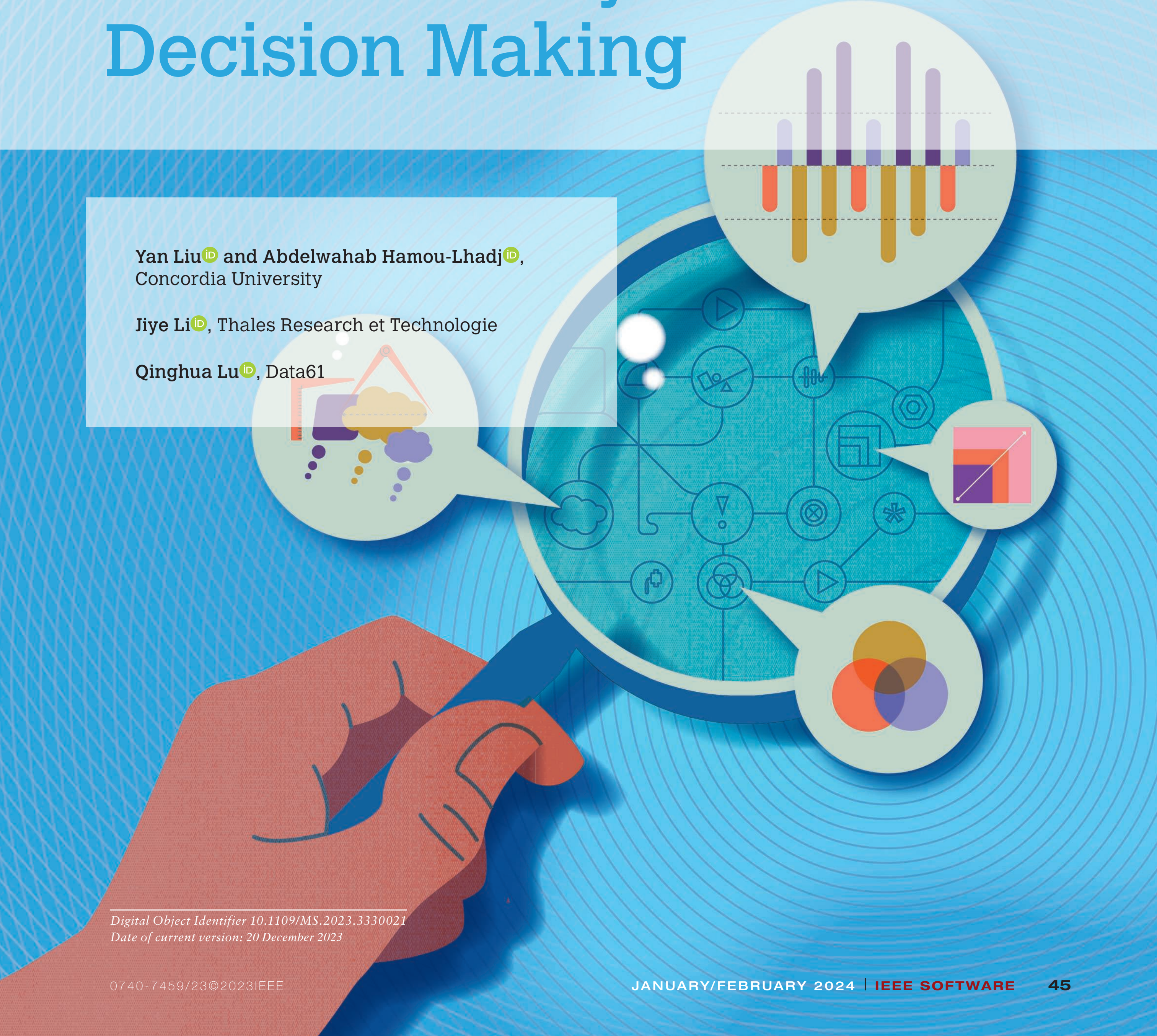
Observability and Explainability for Software Systems Decision Making

Yan Liu and Abdelwahab Hamou-Lhadj, Concordia University

Jiye Li, Thales Research et Technologie

Qinghua Lu, Data61



Digital Object Identifier 10.1109/MS.2023.3330021
Date of current version: 20 December 2023

OBSERVABILITY AND MONITORING

Observability and monitoring both aim to provide insights into the behavior of a system, but they perform in different ways. Monitoring is the task of collecting, analyzing, and using metrics to track a system's progress. Observability is the ability to trace and relate the internal states to behaviors by analyzing the data generated, such as logs, metrics, and traces, across complex and distributed environments. An observable system helps teams understand what is happening to detect and resolve the underlying causes of issues.

SOFTWARE SYSTEMS ARE growing into complex systems under expanding adoptions of the digitization of services, operations, and products in a wide range of domains. It becomes common that software systems operate at the scale of several data centers, hundreds of microservices, thousands of queries per second, millions to hundreds of millions of telemetry time series, and double-digit percent growth monthly as the business grows. Inevitably, software complexity is of high dimensions and of a large scale in almost every dimension. Decision making for such a complex software system is facing the constraints of bounded rationality, including factors of uncertainty, opacity, time limitation, and incomplete information, to ensure successful development, deployment, and maintenance.[1] Therefore, the decision is often of a satisfactory option rather than the best or optimal one. On one hand, bounded rationality can lead to suboptimal decisions, degraded quality, delays, and cost overruns.[2] On the other hand, bounded rationality also fosters opportunities to design processes, architectures, and tools to enhance decision making.[3]

Factors of Decision Making in Complex Software Systems

One of the principles for tackling a complex system, such as a biochemical reaction system, is to obtain observability.[4] *Observability* means the ability to reconstruct a system's internal state from its outputs (see "Observability and Monitoring"). Along with the advancement of technologies on big data storage and processing, massive events, data, and telemetry in both structured and informal formats have been produced and accumulated through the lifecycle of software systems. These data contain comprehensive and valuable information capturing details on relating the internal states to certain behaviors of a software system. The opportunity is on the pathway of combining artificial intelligence (AI), machine learning, and data-mining techniques to handle real-world uncertainties and improve decision making for complex software systems.

While observability generally undertakes the transparent "white-box" approach, explainability relates inputs and outputs and approximates systems as "black box" in a post hoc manner. In a software system with an AI core, explainability has become one of the pillars for trustworthy AI to alleviate users' skepticism, strengthen trust, and promote uptake. Explainable AI is the emerging near-consensus among academics, industries, governments, and civil society groups for developing responsible AI.[5] More broadly, explainability (see "Explainability") has been establishing itself as an important nonfunctional requirement in the context of software systems with complexity and hidden uncertainties.[6] For example, in the domain of digital twins, simulation models become software services that are invoked to compose pipelines for various engineering processes. Under the circumstances that the digital system behavior deviates from the physical system's standards, users require an explanation of the services for the purpose of mission and quality control.

Future Directions

Observability sustains the in-time, continuous, and configurable reconstruction of the internal states of software systems at varying degrees of granularity across a system. Explainability focuses on the approximation of the system's behavior relating to inputs and chosen outputs. Observability and explainability each line up inimitable views on the transparency of software systems. Their synergy addresses not only data-driven solutions but also paves a path to achieve broader missions in software systems, including but not limited to tracing the causality, accountable decision making, automation, and self-organization as well as collaborative intelligence among humans, software systems, and large-scale machine learning.

In fact, synergizing observability and explainability is not an easy task, as there are many challenges involved. Moving forward, possible questions are presented to motivate further exploration in depth and in breadth.

Harnessing the power of observability and explainability benefits reliable and cost-effective decision making in the software development lifecycle.

- *Defining and measuring observability and explainability*: There is no concise definition of *observability* or *explainability*. Different software systems may have different requirements and criteria for observability and explainability. What are the common features of observability and explainability applicable across domains? What are the best practices to measure them?
- *Balancing trade-offs with other quality objectives*: Observability and explainability may have competing objectives to each other and to other quality attributes, such as accuracy, scalability, security, privacy, efficiency, usability, and so on. For example, increasing the amount of data collected may help improve the system's observability and provide more detailed explanations for the system's decision, but it may also incur more complexity, overhead, delays, complexity, and vulnerable risks. How can we evaluate the tradeoff of observability and explainability versus other quality objectives? How can we structure and order data according to the priority of quality objectives?
- *Designing architecture for interactions and feedback*: A complex system often forms a hierarchy of system and subsystems. Interactions among parts at lower levels raise the order of emergence at the higher level without a centralized control. In real-world applications, this may pose the demand of understandable explanation and observability both on a per-device basis and at the aggregation points. In addition, in a system involving intelligence computing, feedback from the environment and people is cardinal to form chains or loops for causality analysis, error correction, and improved accuracy. What are the architecture styles for structuring observable components and feedback chains or loops at various level of a software system? What are the best practices of architecting compositions for observability or explainability?

EXPLAINABILITY

Explainability, in the context of decision making in software systems, refers to the ability to provide clear and understandable reasons behind the decisions, recommendation, and predictions made by the software. Explainability is important for trustworthy interaction with a software system, especially when complex or intelligent algorithms are involved.

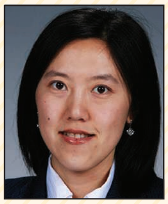
A Road Map of the Special Issue Articles

In this special issue, we begin with a summary of interviews with three specialists who are leading industry observability solutions in the production environment. This interview article[A1] outlines the key challenges and major objectives of industry practices on observability and explainability. It provides a high-level background for understanding the features of observability and explainability. "Explaining Cyberphysical System Behavior With Digital Twins"[A2] presents a model-driven architecture that combines essential components for the explanation of decision making in complex digital twin systems. It shows a starting point of combining domain models, expertise, process, and explainability from established model-driven engineering principles. "Focusing on What Matters: Explaining Quality Tradeoffs in Software-Intensive Systems via Dimensionality Reduction"[A3] presents an architectural approach to explaining the tradeoffs of quality attributes of high dimensionality in software-intensive systems. The explanation is the result of combined analysis of dimension-reduction components.

Furthermore, a vision is depicted in "Explainability With Observation Sharing in Long Collaboration Chains of Automated Systems of Systems"[A4] on how observation is shared in the domain of automated long chains of collaboration with robotics. The linkages between architecture essentials and observable components are well outlined to motivate readers to consider integrated architecture solutions with explainability and observability included. Specifically, "Explaining Black Boxes With a SMILE: Statistical Model-Agnostic Interpretability With Local Explanations"[A5] drills into the property of the explainability of AI models by evaluating how different explanation models react to mutated features. In particular, the article discusses the importance of the stability and trustworthiness of explainable AI models and showcases scenarios with human intuition and that are resilient to adversarial attack.

Overall, the articles in this special issue cover the properties of explainability, architecture essentials, and observable components. The challenges and directions for adoption in practice advocate for continuous effort under this emerging topic.

ABOUT THE AUTHORS


YAN LIU is a tenured associate professor at Concordia University, Montreal, QC H3G 1M8, Canada. Contact her at yan.liu@concordia.ca.

ABDELWAHAB HAMOU-LHADJ is a professor at Concordia University, Montreal, QC H3G 1M8, Canada. Contact him at wahab.hamou-lhadj@concordia.ca.

JIYE LI is a research and technology lead at Thales Research et Technologie, Québec, QC G1P 4P5, Canada. Contact her at jiye.li@thalesgroup.com.

QINGHUA LU is a principal research scientist, Data61, Alexandria, NSW 1435, Australia. Contact her at qinghua.lu@csiro.au.

Acknowledgments
This special issue benefits from the high-quality submissions we received for this theme issue. We express our sincere appreciation to the authors and reviewers for sharing their contributions and expertise. We are also grateful to Editor in Chief Dr. Ipek Ozkaya and the *IEEE Software* team for their support and guidance. Yan Liu and Abdelwahab Hamou-Lhadj are supported by the Natural Sciences and Engineering Research Council of Canada Discovery Grants (individual) program.

Appendix: Related Articles
A1. I. Gorton, L. Fong-Jones, and A. Larsson, "Observability Q&A," *IEEE Softw.*, vol. 41, no. 1, pp. 50–54, Jan./Feb. 2024, doi: 10.1109/MS.2023.3330234.
A2. J. Michael, M. Schwammberger, and A. Wortmann, "Explaining cyberphysical system behavior with digital twins," *IEEE Softw.*, vol. 41, no. 1, pp. 55–63, Jan./Feb. 2024, doi: 10.1109/MS.2023.3319580.
A3. J. Cámara, R. Wohlrab, D. Garlan, and B. Schmerl, "Focusing on what matters: Explaining quality tradeoffs in software-intensive systems via dimensionality reduction," *IEEE Softw.*, vol. 41, no. 1, pp. 64–73, Jan./Feb. 2024, doi: 10.1109/MS.2023.3320689.
A4. P. Daubaris et al., "Explainability with observation sharing in long collaboration chains of automated systems of systems," *IEEE Softw.*, vol. 41, no. 1, pp. 74–86, Jan./Feb. 2024, doi: 10.1109/MS.2023.3320742.
A5. K. Aslansefat, M. Hashemian, M. Walker, M. N. Akram, I. Sorokos, and Y. Papadopoulos, "Explaining

black boxes with a SMILE: Statistical model-agnostic interpretability with local explanations," *IEEE Softw.*, vol. 41, no. 1, pp. 87–97, Jan./Feb. 2024, doi: 10.1109/MS.2023.3321282.

References

1. L. Chazette, W. Brunotte, and T. Speith, "Explainable software systems: From requirements analysis to system evaluation," Re*quirements Eng.*, vol. 27, no. 4, pp. 457–487, 2022, doi: 10.1007/s00766-022-00393-5.
2. H.-M. Chen and R. Kazman, "Architecting ultra-large-scale green information systems," in *Proc. 1st Int. Workshop Green Sustain. Softw. (GREENS)*, 2012, pp. 69–75, doi: 10.1109/GREENS.2012.6224259.
3. L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, "Explaining explanations: An overview of interpretability of machine learning," in *Proc. IEEE 5th Int. Conf. Data Sci. Adv. Analytics (DSAA)*, 2018, pp. 80–89, doi: 10.1109/DSAA.2018.00018.
4. T. Huang, G. Allon, and A. Bassamboo, "Bounded rationality in service systems," *Manuf. Service Operations Manage.*, vol. 15, no. 2, pp. 263–279, 2013, doi: 10.1287/msom.1120.0417.
5. Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, "Observability of complex systems," *Proc. Nat. Acad. Sci. USA*, vol. 110, no. 7, pp. 2460–2465, 2013, doi: 10.1073/pnas.1215508110.
6. C. Smith, C. Babich, and M. Lubrick, *Leadership and Management in Learning Organizations*. Toronto, ON, Canada: eCampusOntario, 2022.

Digital Object Identifier 10.1109/MS.2023.3334609